TR-803 11 August, 1979
DAAG-53-76C-0138

6

A QUADTREE MEDIAL AXIS TRANSFORM.

12

10 Hanan Samet
Computer Science Department ✓
University of Maryland
College Park, MD 20742

Technical rept.,

COMPUTER SCIENCE
TECHNICAL REPORT SERIES

UNIVERSITY OF MARYLAND
COLLEGE PARK, MARYLAND
20742

14 TR-803
DAAG-53-76C-0138

11 August, 1979

6 # A QUADTREE MEDIAL AXIS TRANSFORM.

12 54

10 Hanan/Samet
Computer Science Department
University of Maryland
College Park, MD 20742

9 Technical rept.

ABSTRACT

The skeleton and medial axis transform concepts used in traditional image processing representations are adapted to the quadtree representation. A new data structure termed the Quadtree Medial Axis Transform (QMAT) is defined. An algorithm is presented for the computation of the QMAT of a given quadtree by only examining each BLACK node's adjacent and abutting neighbors. Analysis of the algorithm reveals an average execution time proportional to the number of leaves in the quadtree. Some of the interesting properties of the QMAT vis a vis the quadtree are its compactness and a decreased shift sensitivity.

The support of the Defense Advanced Research Projects Agency and the U.S. Army Night Vision Laboratory under Contract 15 DAAG-53-76C-0138, (DARPA Order-3206) is gratefully acknowledged, as is the help of Kathryn Riley and Pat Young. The author has benefited greatly from discussions with Charles R. Dyer, Shmuel Peleg, and Azriel Rosenfeld.

page -A-
409022

LB

1. Introduction

There are a number of methods of representing images [11] among which are borders, arrays, and skeletons. In [3,4,12-20] attention is focussed on the quadtree [5-8] and its interchangeability with border and array representations. In [3,13,14,18,19] methods for the computation of geometrical properties such as connectivity, perimeter, genus, and distance are presented. In this paper we demonstrate the usefulness of the Chessboard distance transform of [18] in computing the skeleton and medial axis transform [1,2,10,11] of an image represented by a quadtree. This may be useful in performing operations such as propagation, shrinking, and matching [11]. In addition, it leads to a more compact representation of the image which is also less sensitive to shift. We term this representation the QMAT.

Section 2 contains a discussion of skeletons and medial axis transforms and their adaptation to the quadtree. In Section 3 we discuss a number of algorithms for the computation of the QMAT and prove their equivalence. Sections 4 and 5 contain a formal description of the best of these algorithms and an analysis of its execution time. The chosen algorithm is motivated by closely scrutinizing the geometrical properties of the quadtree method of representation. We use a variant of ALGOL 60 [9] in the formal presentation of the algorithm. Reconstruction of a quadtree from its QMAT will be treated in a subsequent paper.

2. Medial Axis Transforms, Quadtrees, and Skeletons

Given an image where the set of points in a certain region are labeled S and the set of points outside of the region are labeled $\bar{S}$ (analoqous to BLACK and WHITE respectively). We say that for a point x and a set V, the distance, according to a suitably defined distance metric, d, from x to the nearest point of V is $d(x,V) = \min\{d(x,y|y \in V\}$. Two points x and y are said to be neighbors if $d(x,y) = 1$. We are interested in a subset of S, say T, such that all elements of T have a distance from $\bar{S}$ which is a local maximum. In other words, for each point in T, no neighboring point in S but not in T has a greater distance from $\bar{S}$. The set of points comprising T is said to constitute a skeletal description of S. As an example, consider the rectangle in Figure 1 whose skeleton consists of line segments labeled a, b, c, d, and e. If we know the points of the skeleton and their associated distance values, then we can reconstruct S exactly. The set of points comprising the skeleton and their associated values is termed the medial axis transform (MAT). Clearly, the MAT of S provides a concise method of defining and representing S. For example, using a Euclidean distance metric [2,11], the MAT corresponding to a circle is a point at its center having a distance value equal to the circle's radius.

Clearly, the definition of the distance metric plays an important role in determining the form of the MAT. The most commonly known distance metric is the Euclidean distance

$$d_E(p,q) = \sqrt{(p_x-q_x)^2 + (p_y-q_y)^2}$$

whose maximal blocks are discs. Two other metrics which are more common in digital picture processing are the Absolute Value distance (also known as the City Block distance)

$$d_A(p,q) = |p_x-q_x| + |p_y-q_y|$$

whose maximal blocks are diamonds, and the Maximal Value distance (also known as the Chessboard distance)

$$d_M(p,q) = \max \{|p_x-q_x|, |p_y-q_y|\}$$

whose maximal blocks are squares. Note that in any case, the MAT determines the entire image although it is clear that a point in the image may lie in more than one maximal block. Figures 2c and 2d show the MATs of the rectangle of Figure 2a using d_A and d_M respectively. Figure 2b shows d_A and d_M for the rectangle, which in this example are identical.

Maximal blocks can be of any size and at any position. Thus they are somewhat unwieldy as primitive elements for representation purposes since the process of determining them may be complex. The quadtree approach to image representation is an attempt to exploit the maximal block concept in a more systematic manner. Given a 2^n by 2^n array of unit pixels, we repeatedly subdivide the array into quadrants, subquadrants,... until we obtain blocks (possibly single pixels) which consist entirely of a single value (e.g., gray level). This process is represented by a tree of out degree 4 in which the root node represents the entire array, the four sons of the root node represent the quadrants, and the terminal nodes correspond to those blocks of the array for which no further subdivision is necessary. The nodes at level k (if any) represent blocks of size $2^k x 2^k$ and are often referred to as nodes of size 2^k. Thus a node at level Ø corresponds to a single pixel in the image, while a node at level n is the root of the quadtree. For example, Figure 3b is a block decomposition of the region in Figure 3a while Figure 3c is the corresponding quadtree. In general, we will be dealing with two values 1 and Ø where BLACK and WHITE square nodes in the tree represent blocks consisting entirely of 1's and Ø's respectively. Circular nodes, also termed GRAY nodes, denote non-terminal nodes.

In [18,19] the concept of distance is applied to a quadtree. In particular, it was shown in [18] that the Chessboard distance metric is especially suitable for a quadtree since it has the property that given a point P, the set of points q such that $d(p,q) \leq t$ is a square. The Chessboard distance transform for a quadtree, DIST, was defined as a function that yields for each BLACK block in the quadtree the Chessboard distance from the center of the block to the nearest point which is on a BLACK-WHITE border. More formally, letting x be the center of a BLACK block b, z be a point on the border of the WHITE block w, we have

$$F(b,w) = \min_{z} d(x,z)$$

$$DIST(b) = \min_{w} F(b,w)$$

We also say that DIST of a WHITE block is zero and that the border is BLACK for the purpose of the computation of F and DIST.

We are now ready to define the Quadtree Medial Axis Transform (QMAT). We first define the Quadtree Skeleton. Let the set of BLACK blocks in the image be denoted by B. For each BLACK block, b_i, let $S(b_i)$ be the part of the image spanned by a square with side width $2*DIST(b_i)$ centered about b_i. The Quadtree Skeleton consists of the set, T, of BLACK blocks satisfying the following properties:

(1) $area(B) = UNION(S(t_i))$

(2) for any $t_j \in T \nexists b_k \in B \ (k \neq j) \ni S(t_j) \subseteq S(b_k$

(3) $\forall b_i \in B \ \exists t_j \in T \ni S(b_i) \subseteq S(t_j)$

Property (1) insures that the entire image is spanned by the skeleton. Property (2) is termed the subsumption property and we say that b_j is subsumed by b_k when $S(b_j) \subseteq S(b_k)$. Property (2) means that the elements of the Quadtree Skeleton are the blocks with the largest distance transform values. Note that it is not the same as saying that

$\nexists t_k \in T\ (k \neq j) \ni S(t_j) \subseteq S(t_k$ as will be seen below. Property (3) insures that no block in B and not in T requires more than one element of T for its subsumption--e.g., one half of the block is subsumed by one element of T and the other half is subsumed by another element of T.

<u>Theorem 1</u>: The Quadtree Skeleton of an image is unique.

<u>Proof</u>: Assume that the Quadtree Skeleton is not unique. Let T_1 and T_2 both be Quadtree Skeletons of the same image--i.e., $B = \text{UNION}(S(t_{1i}))\ t_{1i} \in T_1$ and $B = \text{UNION}(S(t_{2j})\ t_{2j} \in T_2$. Assume, without loss of generality, that $\exists t_{1k} \in T_1 \ni t_{1k} \notin T_2$. Therefore, by property (3) $\exists t_{2\ell} \in T_2\ (t_{2\ell} \neq t_{1k}) \ni S(t_{1i}) \subseteq S(t_{2\ell})$. However, this contradicts property (2) which stipulates that for any $t_{1i} \in T_1\ \nexists b_i \in B\ (b_i \neq t_{1i}) \ni S(t_{1i}) \subseteq S(b_i)$. Hence, the Quadtree Skeleton of an image is unique.

Q.E.D.

The QMAT of an image is the quadtree whose BLACK nodes correspond to the BLACK blocks comprising the Quadtree Skeleton and their associated Chessboard distance transform values. All remaining nodes in the QMAT are WHITE and GRAY with distance value zero. For example, Figure 3d contains the Chessboard distance transform corresponding to the region given in Figure 3a. Figures 3c and 3f contain the block and tree representations respectively of the QMAT of Figure 3a.

We now make the following observations with the aid of Figure 3. The squares spanned by the Chessboard distance transform of the blocks of the QMAT are not necessarily disjoint. For example, block 11 is subsumed by both blocks 1 and 15. Recall that the subsumption property (i.e., property (2)) means that the elements of the QMAT are the blocks with the largest distance transform values. For example, the QMAT consists of blocks 1, 11, and 15 rather than blocks 1, 12, and 15 since block 12 is subsumed by block 11. The latter result would have been legal had we defined the Quadtree Skeleton as the set, T, of BLACK blocks such that area(B) = $\mathrm{UNION}(S(t_i))\,t_i \in T$ and for any $t_j \in T \nexists\, t_k \in T$, $k \neq j$, such that $S(t_j) \subseteq S(t_k)$. Note that such a definition would lead to a QMAT which contains more nodes (e.g., WHITE node N in Figure 3f would be replaced by a GRAY node having BLACK son 12 and WHITE sons 30, 31, and 32). The fact that the border of the image is assumed to be BLACK results in minimizing the number of nodes in the QMAT. Without this assumption, block 1 would be of radius 4 and would not lead to the subsumption of blocks 2, 3, 4, 8, 9, and 10. Note that blocks 5 and 11 are subsumed by block 15 anyway so that their subsumption is not dependent on our assumption.

Before proceeding any further it is appropriate to make a few additional comments about property (3) of the Quadtree Skeleton definition. This property does not yield a minimal set of blocks. For example, in the image of Figure 4a, property (3) requires that the Quadtree Skeleton contain blocks 5, 14, and 15 while in actuality blocks 5 and 15 are sufficient since together they subsume block 14. Thus if we were interested in a minimal set of blocks we would modify property (3) as in (3') below:

(3') $\nexists t_j \in T \ni S(t_j) \subseteq \text{UNION}(S(t_k)) \quad t_k \neq t_j$

The reason we do not use the definition of a Quadtree Skeleton which yields the minimal set of blocks is two-fold. First, by virtue of the definition of the QMAT, the tree size of the QMAT would be unaffected by using property (3') instead of property (3) since the only difference is that the additional blocks are represented by BLACK nodes rather than WHITE nodes (e.g., node 14 in Figures 4b and 4c). That this is always true can be seen easily by observing that for a node to be extraneous by virtue of property (3'), it must be subsumed by its neighbors which must themselves be BLACK. Thus the extraneous node when represented by a WHITE node cannot be merged with its neighbors to yield a larger node and must remain a part of the QMAT. Second, as will be seen in Section 3, the QMAT creation algorithm is considerably simpler when property (3) is used.

The QMAT representation can be used as an alternative data structure for the representation of an image. In particular, it has the property that for any image it requires at most as many nodes as the quadtree. This is obvious when we recall that each node in the QMAT corresponds to one or more nodes of the quadtree and that each member of the Quadtree Skeleton is a node in the quadtree. Of course, the QMAT does require that the DIST value be stored with each node. As an example of the savings in storage, consider the image in Figure 3a. The QMAT, shown in Figure 3f, requires 17 nodes while the quadtree, shown in Figure 3c, requires 57 nodes.

An interesting property of the QMAT is that there is a class of images for which it requires a minimum number of nodes regardless of the image resolution. Clearly, if the image is all WHITE or all BLACK, then both the quadtree and the QMAT require a single node. However, when the image consists entirely of BLACK blocks with the exception of a row and column of WHITE blocks of equal size adjacent to two touching sides, then the advantage of the QMAT over the quadtree in terms of space utilization is at a maximum. For example, consider the image in Figure 5a and its quadtree and QMAT in Figures 5b and 5c,respectively. The quadtree requires 45 nodes while the QMAT requires only 5 nodes. In fact, for any image of such a shape the QMAT requires only 5 nodes while the quadtree requires a number of nodes which depends on the maximum level of the tree. The exact number of nodes required for such a quadtree of level n can be obtained by use of the following recurrence relations. Assume, without loss of generality, an image similar to that of Figure 5a (i.e., the largest block is in the NW quadrant). Let t_i denote the number of nodes in a quadtree of level i and $s_{i=1}$ be the contribution made by the NE and SW quadrants of a quadtree of level i.

$$t_i = \begin{cases} 1 & i = \emptyset \\ 1 + 1 + 2 \cdot s_{i-1} + t_{i-1} & i \geq 1 \end{cases}$$

$$s_i = \begin{cases} 1 & i = \emptyset \\ i + 2 + 2 \cdot s_{i-1} & i \geq 1 \end{cases}$$

It can be easily shown that these relations have the following solutions:

$$s_i = 2^{i+2} - 3$$

$$t_i = 2^{i+3} - 4 \cdot i - 7$$

To see this, we observe that

$$s_i = 3 \cdot \sum_{j=0}^{i-1} 2^j - 2^{i+1} = 3 \cdot (2^i - 1) + 2^i = 2^{i+2} - 3$$

and substituting into t_i we have

$$t_i = 2 + 2 \cdot (2^{i+1} - 3) + t_{i-1} = 4 \cdot (2^i - 1) + t_{i-1}$$

Solving for t_i we get

$$t_i = 4 \sum_{j=1}^{i} (2^j - 1) + 1 = 4 \cdot (2^{i+1} - 2 - i) + 1 = 2^{i+3} - 4 \cdot i - 7$$

Thus, for a quadtree of level n, the number of nodes that can be saved by using the QMAT representation is $2^{n+3} - 4 \cdot n - 12$. For example, for n=3, the difference is 40 nodes--i.e., a reduction by a factor of 15.

The QMAT representation also has the property that the number of nodes necessary to represent an image is not as shift-sensitive as is the quadtree. For example, when the image of Figure 6a is shifted by one unit to the right yielding Figure 6d, its quadtree gets considerably larger. In particular, Figure 6b contains 17 nodes while Figure 6e, the quadtree corresponding to the shifted image, contains 49 nodes. However, the QMAT is not as sensitive to shifts since it always requires a number of nodes less than or equal to those contained in the quadtree. In Figure 6, the QMAT of Figure 6a, given in Figure 6c, is identical to the quadtree. However, the QMAT of the shifted image, given in Figure 6f, is considerably smaller than its corresponding quadtree as well as the QMAT of the image prior to the shift (i.e., 9 nodes vs. 17 nodes). As another example, consider the image of Figure 7a which has a minimal nontrivial QMAT in terms of the number of nodes (i.e., 5 nodes). Figure 7d is the result of shifting the image of Figure 7a by one unit to the right. Note that the new QMAT given in Figure 7f requires more nodes than the one corresponding to the unshifted image given in Figure 7c (i.e., 21 nodes versus 5 nodes). However, this number is less than the number of nodes in the shifted quadtree as shown in Figure 7e (i.e., 21 nodes for the QMAT versus 41 nodes for the shifted quadtree). Thus we see that the compactness of the QMAT is also preserved when the image is subjected to shifts.

3. Algorithms for the Computation of the QMAT

Properties (2) and (3) of Section 2 suggest the following simple two step algorithm (termed A) for determining the QMAT. At the end of the algorithm T contains the BLACK blocks comprising the QMAT.

Algorithm A:

(1) Sort the BLACK blocks in increasing order by value of their Chessboard distance transform forming the set T--i.e., $\text{DIST}(t_i) \leq \text{DIST}(t_{i+1}) \quad t_i \in T$

(2) Starting with i=1: For each $t_i \in T \ni \exists\ t_j (i<j)$ and $S(t_i) \subseteq S(t_j)$, then remove t_i from T.

From a computation standpoint Algorithm A is quite complex since it involves sorting the BLACK blocks as well as examining whether or not a block is subsumed by the remaining blocks. Instead, we use an algorithm, termed Algorithm B, which traverses the quadtree in postorder (i.e., the sons of a node are visited first) and determines for each node corresponding to a BLACK block, say P, whether $S(P) \subseteq S(Q)$. Q is one of P's eight neighbors in the N, NE, E, SE, S, SW, W, and NW directions. In general, whenever a BLACK block is subsumed by one of its neighbors, then it appears in the QMAT as a WHITE block. Once all the sons of a GRAY node have been processed, then if they all correspond to WHITE blocks, then the node is changed to correspond to a WHITE block (e.g., GRAY node N of Figure 3c having sons 30, 12, 31, and 32 is changed to correspond to a WHITE block in Figure 3f).

At this point it is appropriate to examine the notion of subsumption in a more rigorous manner. Given adjacent nodes Q corresponding to BLACK blocks appearing at levels L_P and L_Q respectively in the quadtree such that $L_P > L_Q$, and letting $D(P,Q)=DIST(Q) = 2\uparrow(L_Q-1) - 2\uparrow(L_P-1)$, then P is said to be subsumed by Q if $D(P,Q)=DIST(P)$. It should be clear that $D(P,Q)$ cannot be greater than DIST(P) since this would contradict the definition of the Chessboard distance transform (i.e., P would have a closer BLACK-WHITE border point than Q although being constrained by the value of D(P,Q) to be entirely contained in the square of side width 2*DIST(Q) centered at Q). Clearly, when $D(P,Q) < DIST(P)$, P is not subsumed by Q.

When $D(P,Q)=DIST(P)$, there are two cases to consider. If $DIST(P)=2\uparrow(L_P-1)$, then P is adjacent to the outer border of S(Q) and thus no BLACK blocks can be subsumed by P (e.g., BLACK block 9 in Figure 36 is adjacent to the outer border of the square spanned by block 1). Thus changing block P from BLACK to WHITE will not affect the detection of subsumption of other nodes.

However, if $DIST(P) > 2\uparrow(L_P-1)$ (i.e., the second case to be considered when $D(P,Q)=DIST(P)$), then P is not adjacent to the outer border of S(Q). This means that some blocks which are subsumed by Q can only be detected by virtue of being subsumed by P since they are not adjacent to Q. Denote these blocks by S(P,Q). It can be shown that all elements of

S(P,Q) satisfy the following properties:

(1) Q is of larger size than P.

(2) Each element of S(P,Q) is smaller in size than P.

(3) If Q is adjacent to P along side S of the BLACK block corresponding to P, then S(P,Q) is equal to the blocks subsumed by the opposite side, denoted by OPSIDE(S), of P's block.

(4) If Q abuts the corner formed by sides S and T of the BLACK block corresponding to P, then S(P,Q) is equal to the blocks subsumed by sides OPSIDE(S) and OPSIDE(T) of P's block.

Properties (1)-(4) imply that all elements of S(P,Q) are in the space spanned by FATHER(P)--i.e., they are in the region spanned by the brothers of P. This means that an algorithm that processes a GRAY son prior to its BLACK or WHITE brothers insures that the QMAT is formed by examining blocks for subsumption according to increasing size (i.e., smaller size first). As soon as a BLACK block is determined to be subsumed by its neighbor, its DIST and NODETYPE fields are changed to zero and WHITE respectively. This leads to the following result.

<u>Lemma 1</u>: Both Algorithms A and B satisfy the definition of a Quadtree Skeleton and the QMAT of an image.

Proof: Algorithm A clearly satisfies properties (1)-(3) of the definition since its steps are equivalent to the definition. To show that Algorithm B meets our requirements is slightly more complex. Properties (1) and (3) are satisfied since Algorithm B starts with the QMAT and the quadtree being identical and then systematically removes nodes whose corresponding blocks are subsumed by others. Satisfaction of property (2) is shown as follows. Algorithm B is based on the principle that each block is subsumed by a neighboring block. It examines each adjacency and removes a BLACK block from the skeleton if it is subsumed by an adjacent block in the skeleton (i.e., one that has not yet been removed by virtue of being subsumed by yet another larger adjacent block). Properties (1)-(4) of the case when $D(P,Q)=DIST(P)$ and $DIST(P) > 2\uparrow(L_P-1)$ and the fact that a GRAY son is processed before its BLACK and WHITE brothers insure that no block is removed from the skeleton before blocks that are subsumed by it. Thus we see that no block in the QMAT is subsumed by another block in the quadtree. Recall from Section 2 that this is a stronger statement than not being subsumed by another node in the QMAT.

Q.E.D.

Theorem 2: Algorithms A and B are equivalent.

Proof: By Lemma 1 both Algorithms A and B compute the Quadtree Skeleton. Theorem 1 indicates that the Quadtree Skeleton of an image is unique and our result follows.

Q.E.D.

The equivalence of Algorithms A and B can also be seen by observing that they both start with the smallest BLACK blocks and attempt to determine if they are subsumed by other BLACK blocks. The key to the superiority of Algorithm B is that no sorting is required and also that blocks that cannot possible subsume one another are not checked for subsumption--i.e., Algorithm B only examines a maximum of eight neigboring blocks while Algorithm A examines all possible larger sized BLACK blocks. Also note the simplicity of Algorithm B that results from using property (3) rather than (3') in the definition of a Quadtree Skeleton, since each block in the original image can only be subsumed in its entirety. Thus there is no need to examine whether a node is subsumed by a set of other nodes (e.g., node 14 of Figure 4a is subsumed by nodes 5 and 15).

4. Formal Statement of the Algorithm

Prior to describing our algorithm it is useful to define our representation as well as some elementary operations. Let each node in a quadtree be stored as a record containing seven fields. The first five fields contain pointers to the node's father and its four sons, labeled NW, NE, SE, and SW. Given a node P and a son I, these fields are referenced as FATHER(P) and SON(P,I) respectively. At times it is useful to use the function SONTYPE(P) where SONTYPE(P)=Q iff SON(FATHER(P),Q)=P. The sixth field, NODETYPE, describes the contents of the block of the image which the node represents--i.e., BLACK, WHITE, or GRAY. The seventh field, DIST, indicates the value of the Chessboard distance transform for the node. This field is only meaningful for BLACK nodes. WHITE and GRAY nodes are said to have a DIST value of zero. Note that this is different from the concept of node distance--i.e., for a node at level i, n-i FATHER links must be ascended to reach the root of the tree.

The four sides of a node's block are called its N, E, S, and W sides. They are also termed its boundaries. The expression of operations involving a block's quadrants and boundaries is facilitated by the following predicates and functions. ADJ(B,I) is true if and only if quadrant I is adjacent to boundary B of the node's block; e.g., ADJ(W,SW) is true. REFLECT(B,I) yields the quadrant which is adjacent to quadrant

I along boundary B of the block represented by I; e.g., REFLECT(N,SW)=NW, REFLECT(E,SW)=SE, REFLECT(S,SW)=NW, and REFLECT(W,SW)=SE. CSIDE(B) is a side adjacent to side B in the clockwise direction; e.g., CSIDE(N)=E. COMMONSIDE(Q1,Q2) indicates the boundary of a block which is common to quadrants Q1 and Q2 (if Q1 and Q2 are not adjacent quadrants, then the value of COMMONSIDE(Q1,Q2) is undefined); e.g., COMMONSIDE(NW,SW) =W. QUAD(S1,S2) is the quadrant bounded by boundaries S1 and S2 (if S1 and S2 are not adjacent boundaries, then QUAD(S1,S2) is undefined); e.g., QUAD(N,W)=NW. OPQUAD(Q) is a quadrant which is diagonally facing quadrant Q; e.g., OPQUAD(NW)=SE, OPQUAD(NE)=SW. Figure 8 shows the relationship between the quadrants of a node and its boundaries.

The algorithm that is described is different from Algorithm B in that it has been modified to avoid having to distinguish between GRAY sons and their BLACK and WHITE brothers. Instead, whenever a BLACK block, say P, has been found to be subsumed by an adjacent BLACK block, say Q, then P's NODETYPE field is changed to WHITE but its DIST field is left alone. This insures that application of the QMAT algorithm to any of P's yet unprocessed GRAY brothers will result in their subsumption by P if appropriate. Note that when Q is a genuine WHITE block, D(P,Q) is negative since DIST(Q) is zero, and thus P can't be subsumed by Q--i.e., D(P,Q) < DIST(P). Once all of a GRAY node's sons have been processed, a check is made if they all correspond to WHITE blocks. If yes, then they and their father are replaced by a node having NODETYPE and DIST field values WHITE and zero respectively. Otherwise, the DIST field of any son corresponding to a WHITE block is set to zero.

The main procedure is termed QMAT and is invoked with a pointer to the root of the quadtree representing the image and an integer corresponding to the log of the diameter of the image (e.g., n for a 2^n by 2^n image array). We assume that each block's distance has already been computed by a method such as that described in [14]. QMAT traverses the tree and controls the examination of the eight neighbors of each BLACK node. Note that our algorithm results in transforming the original quadtree to a QMAT by overwriting the original quadtree. This is not necessary. An alternative algorithm would create copies of the nodes while forming the QMAT. In fact, the only modification to our algorithm is to create a copy of each node prior to examining its neighbors.

Procedure FIND_NEIGHBOR locates a neighboring node of greater or equal size along a specified side (e.g., N, E, S, or W). If the node is on the edge of the image, then no neighbor exists in the specified direction and NULL is returned (e.g., the western neighbor of node 1 in Figure 3b).

If the node is not on the edge of the image and no neighboring BLACK or WHITE node exists satisfying our size criteria, then a pointer to a GRAY node of equal size is returned (e.g., the eastern border of node 1 in Figure 3b). Procedure FIND_CORNER is analogous to FIND_NEIGHBOR and locates a neighboring node of greater or equal size along a corner (e.g., NW, NE, SE, or SW). For example, node 7 is the NE neighbor of node 15 in Figure 3b.

As an example of the application of the algorithm, consider the region given in Figure 3a. Figure 3b is the corresponding block decomposition while Figure 3c is its quadtree representation. All of the BLACK nodes have labels ranging from 1 to 20 while the WHITE nodes have labels ranging from 21 to 43. The GRAY nodes have labels ranging between A and N. The BLACK nodes have been labeled in the order in which their subsuming adjacencies were explored by procedure QMAT. Figure 3d contains the Chessboard distance transform corresponding to Figure 3b. Figures 3e and 3f contain the block decomposition of the QMAT and the quadtree representation of the QMAT corresponding to Figure 3b respectively.

```
procedure QMAT(P,LEVEL);
/*  convert the quadtree rooted at node P representing a
    $2^{LEVEL}$ by $2^{LEVEL}$ image to its quadtree medial axis
    transform  */
begin
    node P,Q;
    integer L, LEVEL;
    quadrant I;
    side S;
    if BLACK(P) then
        begin
            for S in {N,E,S,W} do
                begin
                    FIND_NEIGHBOR(P,S,Q,L←LEVEL);
                    if not NULL(Q) and
                       DIST(Q)-2↑(L-1)-2↑(LEVEL-1)=DIST(P) then
                        begin /* P is subsumed by its neighbor Q */
                            NODETYPE(P)←WHITE;
                            return;
                        end;
                    FIND CORNER(P,QUAD(S,CSIDE(S)),Q,L←LEVEL);
                    if not NULL(Q) and
                       DIST(Q)-2↑(L-1)-2↑(LEVEL-1)=DIST(P) then
                        begin /* P is subsumed by its neighbor Q */
```

```
                                NODETYPE(P)←WHITE;
                                return;
                            end;
                    end;
    end
    else if GRAY(P) then
        begin
            for I in {NW,NE,SW,SE} do QMAT(SON(P,I),LEVEL-1);
            if WHITE(SON(P,NW)) and WHITE(SON(P,NE)) and
               WHITE(SON(P,SE)) and WHITE(SON(P,SW)) then
                begin /* merge the four sons */
                    NODETYPE(P)←WHITE;
                    for I in {NW,NE,SE,SW} do
                        begin
                            RETURNTOAVAIL (SON(P,I));
                            SON(P,I)←NULL;
                        end;
                end;
            else
                begin
                    for I in {NW,NE,SE,SW} do
                        begin
                            if WHITE(SON(P,I)) then DIST(SON(P,I))←Q;
                        end;
                end;
        end;
end;  /* WHITE nodes are left alone */
```

```
procedure FIND_NEIGHBOR(P,S,Q,L)
/*  given node P, return in Q the node which is adjacent to
    side S of node P.  L denotes the level of the tree at
    which node P is initially found and the level of the tree
    at which Q is finally found  */
begin
    node P;
    reference node Q;
    side S;
    reference integer L;
    L←L+1;
    if not NULL(FATHER(P)) and ADJ(S,SONTYPE(P)) then
        /*  find a common ancestor  */
        FIND_NEIGHBOR(FATHER(P),S,Q,L)
    else Q←FATHER(P);
    /* follow reflected path to locate the neighbor */
    if not NULL(Q) and GRAY(Q) then
        begin
            Q←SON(Q,REFLECT(S,SONTYPE(P)));
            L←L-1;
        end;
end;
```

```
procedure FIND_CORNER(P,C,Q,L);
/*  given node P, return in Q the node which is adjacent to
    corner C of node P.  L denotes the level of the tree at
    which node P is initially found and the level of the tree
    at which Q is finally found  */
begin
    node P;
    reference node Q;
    quadrant C;
    reference integer L;
    L←L+1;
    if not NULL(FATHER(P)) and SONTYPE(P)≠OPQUAD(C) then
        if SONTYPE(P)=C then FIND_CORNER(FATHER(P),C,Q,L)
        else FIND_NEIGHBOR(FATHER(P),COMMONSIDE(SONTYPE(P),C),Q,L)
    else Q←FATHER(P);
    /*  follow opposite path to locate the neighbor  */
    if not NULL(Q) and GRAY(Q) then
        begin
            Q←SON(Q,OPQUAD(SONTYPE(P)));
            L←L-1;
        end;
end;
```

5. Analysis

The running time of the QMAT computation algorithm is measured by the number of nodes that are visited and by the size of the quadtree. For each BLACK node we must visit a minimum of one neighbor and a maximum of eight neighbors of greater than or equal size in order to determine whether the block corresponding to the node is subsumed--i.e., contained in a square centered at its neighbor (e.g., block 14 is subsumed by block 15 in Figure 3b). Clearly, for each BLACK node, the worst case in terms of the number of nodes that must be visited arises when the neighbor that is being sought is of equal size (e.g., the NE neighbor of block 11 in Figure 3b--i.e., block 5). Thus we only need to analyze the amount of work performed by procedures FIND_NEIGHBOR, FIND_CORNER, and QMAT. Our analysis assumes a 2^n by 2^n random image--i.e., a BLACK node is equally likely to appear in any position and level in a quadtree. The analysis closely parallels that performed in [14] for the Chessboard distance transform.

Lemma 2: The average of the maximum number of nodes visited by each invocation of FIND_NEIGHBOR is less than 4.

Proof: Given a node P at level i and a horizontal or vertical direction S, there are $2^{n-i} \cdot (2^{n-i}-1)$ possible positions for node P and a neighbor at level i and direction S. Of these

$2^{n-i}\cdot(2^{n-i}-1)$ neighbor pairs, $2^{n-i}\cdot 2^{0}$ have their nearest common ancestor at level n, $2^{n-i}\cdot 2^{1}$ at level n-1,..., and $2^{n-i}\cdot 2^{n-i-1}$ at level i+1. For each node at level i having a common ancestor at level j, the maximum number of nodes that will be visited by FIND_NEIGHBOR is (j-i) + (j-i) = 2·(j-i). Assuming that node P is equally likely to occur at any level i and at any of the $2^{n-i}\cdot(2^{n-i}-1)$ positions at level i, then the average of the maximum number of nodes visited by FIND_NEIGHBOR is

$$\frac{\sum_{i=0}^{n-1}\sum_{j=i+1}^{n} 2^{n-i}\cdot 2^{n-j}\cdot 2(j-i)}{\sum_{i=0}^{n-1} 2^{n-i}\cdot(2^{n-i}-1)} \tag{1}$$

(1) can be rewritten to yield

$$\frac{\sum_{i=0}^{n-1}\sum_{j=0}^{n-1-i} 2^{2n-2i-j}(j+1)}{\sum_{i=1}^{n} 2^{i}(2^{i}-1)} \tag{2}$$

The numerator of (2) can be simplified as follows:

$$\sum_{i=0}^{n-1}\sum_{j=0}^{n-1-i} 2^{2n-2i-j}(j+1) = \sum_{i=0}^{n-1} 2^{2n-2i}\sum_{j=0}^{n-1-i}\frac{j}{2^{j}} + \sum_{i=0}^{n-1} 2^{2n-2i}\sum_{j=0}^{n-1-i}\frac{1}{2^{j}} \quad ($$

But
$$\sum_{j=0}^{n-1-i}\frac{j}{2^{j}} = 2 - \frac{n+1-i}{2^{n-1-i}} \tag{4}$$

Also
$$\sum_{j=0}^{n-1-i}\frac{1}{2^{j}} = 2\left(1 - \frac{1}{2^{n-i}}\right) \tag{5}$$

Substituting (4) and (5) into (3) yields

$$\sum_{i=0}^{n-1} 2^{2n-2i}\left(2 - \frac{n+1-i}{2^{n-1-i}} + 2\left(1 - \frac{1}{2^{n-i}}\right)\right)$$

$$= \sum_{i=0}^{n-1} \left(2^{2n+2-2i} - 2^{n+1-i}(n+1) + 2^{n+1-i}\cdot i - 2^{n+1-i}\right)$$

$$= 2^{2n+2} \sum_{i=0}^{n-1} \frac{1}{2^{2i}} - 2^{n+1}\cdot(n+2) \sum_{i=0}^{n-1} \frac{1}{2^{i}} + 2^{n+1} \sum_{i=0}^{n-1} \frac{i}{2^{i}}$$

$$= 2^{2n+2}\cdot \frac{4}{3} \cdot \left(1-\frac{1}{2^{2n}}\right) - 2^{n+1}\cdot(n+2)\cdot 2\left(1- \frac{1}{2^{n}}\right) + 2^{n+1}\left(2- \frac{n+1}{2^{n-1}}\right)$$

$$= \frac{4}{3}\, 2^{2n+2} - \frac{16}{3} - (n+2)\cdot 2^{n+2} + 4\cdot(n+2) + 2^{n+2} - 4(n+1)$$

$$= \frac{4}{3}\, 2^{2n+2} - (n+1)\cdot 2^{n+2} - \frac{4}{3} \qquad (6)$$

The denominator of (2) can be simplified as follows:

$$\sum_{i=1}^{n} 2^{i}(2^{i}-1) = \sum_{i=0}^{n} (2^{2i}-2^{i})$$

$$= \sum_{i=0}^{n} 4^{i} - \sum_{i=0}^{n} 2^{i}$$

$$= \frac{4^{n+1}-1}{3} - (2^{n+1}-1)$$

or

$$\sum_{i=1}^{n} 2^{i}(2^{i}-1) = \frac{1}{3}(2^{2n+2}-3\cdot 2^{n+1}+2) \qquad (7)$$

Substituting (6) and (7) into (2) yields

$$\frac{\frac{4}{3}\cdot 2^{2n+2} - (n+1)\cdot 2^{n+2} - \frac{4}{3}}{\frac{1}{3}(2^{2n+2} - 3\cdot 2^{n+1}+2)} = 4 - \frac{3(n-1)\cdot 2^{n+2}+12}{2^{2n+2}-3\cdot 2^{n+1}+2}$$

$$\simeq 4 \text{ as } n \text{ gets large}$$

$$< 4$$

Q.E.D.

Lemma 3: The average of the maximum number of nodes visited by each invocation of FIND_CORNER is less than $\frac{16}{3}$.

Proof: Given a node P at level i and a diagonal direction S, there are $(2^{n-i}-1)^2$ possible positions for node P and a neighbor at level i in direction S. Of these $(2^{n-i}-1)^2$ neighbor pairs, $4^0\cdot(2\cdot(2^{n-i}-1)-1)$ have their nearest common ancestor at level n, $4^1(2\cdot(2^{n-i-1}-1)-1)$ at level n-1,... and $4^{n-i-1}\cdot(2\cdot(2^{n-i-(n-i-1)}-1)-1)$ at level i+1. In order to see this, consider Figure 9 where a grid is shown for n=3. If all BLACK and WHITE nodes are at level Ø, then for a neighbor in the NE direction we see that nodes along the fifth row and fourth column have their nearest common ancestor at level 3 (i.e., 13 nodes labeled 1-13). Continuing the process for the NW, NE, SW, and SE quadrants of Figure 9 we find that all neighbor pairs contained exclusively within these quadrants have their nearest common ancestor at a level ≤ 2. In particular, for the NW quadrant, nodes along the third row and second column have their nearest common ancestor at level 2 (i.e., 5 nodes labeled 14-18). The NE, SW, and SE quadrants are analyzed in a similar manner. This process is applied to the four subquadrants of the quadrants to obtain the neighbor pairs whose nearest common ancestor is at level 1. Note that we had to consider every row in the image when analyzing diagonal neighbor pairs whereas we only needed to consider one row or column when analyzing neighbor pairs in the N, E, S, and W directions. This is necessary because for diagonal neighbors, each row in the image has a different number of neighbor pairs with a common ancestor at a given level while this number is constant for each row or column when considering neighbor pairs in the horizontal and vertical directions.

For each node P at level i having a common ancestor at level j, the maximum number of nodes that will be visited by FIND_CORNER is $(j-i)+(j-1) = 2\cdot(j-i)$. Assuming that node P is equally likely to occur at any level i and at any of the $(2^{n-i}-1)^2$ positions at level i, then the average of the maximum number of nodes visited by FIND_CORNER is

$$\frac{\sum_{i=0}^{n-1}\sum_{j=i+1}^{n} 4^{n-j}\cdot(2\cdot(2^{n-i-(n-j)}-1)-1)\cdot 2\cdot(j-i)}{\sum_{i=0}^{n-1}(2^{n-i}-1)^2}$$

$$\frac{\sum_{i=0}^{n-1}\sum_{j=i+1}^{n}(2^{2n-j-i+2}-3\cdot 2^{2n-2j+1})(j-i)}{\sum_{i=0}^{n-1}(2^{n-i}-1)^2} \qquad (8)$$

(8) can be rewritten to yield

$$\frac{\sum_{i=0}^{n-1}\sum_{j=0}^{n-1-i}(2^{2n-2i+1-j}-3\cdot 2^{2n-2i-1-2j})(j+1)}{\sum_{i=1}^{n}(2^{i}-1)^{2}} \qquad (9)$$

The numerator of (9) can be simplified as follows

$$\sum_{i=0}^{n-1}\sum_{j=0}^{n-1-i}(2^{2n-2i+1-j}-3\cdot 2^{2n-2i-1-2j})(j+1)$$

$$= \sum_{i=0}^{n-1}(2^{2n-2i+1}(\sum_{j=0}^{n-1-i}\frac{j}{2^j}+\sum_{j=0}^{n-1-i}\frac{1}{2^j})-3\cdot 2^{2n-2i-1}(\sum_{j=0}^{n-1-i}\frac{j}{2^{2j}}+\sum_{j=0}^{n-1-i}\frac{1}{2^{2j}}) \qquad (10)$$

But $$\sum_{j=0}^{n-1-i}\frac{j}{2^j} = 2 - \frac{n+1-i}{2^{n-1-i}} \qquad (11)$$

$$\sum_{j=0}^{n-1-i}\frac{1}{2^j} = 2(1-\frac{1}{2^{n-i}}) \qquad (12)$$

$$\sum_{j=0}^{n-1-i}\frac{j}{2^{2j}} = \frac{1}{9}(4 - \frac{3(n-1-i)+4}{2^{2n-2-2i}}) \qquad (13)$$

$$\sum_{j=0}^{n-1-i}\frac{1}{2^{2j}} = \frac{4}{3}(1 - \frac{1}{2^{2n-2i}}) \qquad (14)$$

Substituting (11), (12), (13), and (14) into (10) yields

$$\sum_{i=0}^{n-1}(2^{2n-2i+1}(2 - \frac{n+1-i}{2^{n-1-i}} + 2(1 - \frac{1}{2^{n-1}}))-3\cdot 2^{2n-2i-1}$$

$$\cdot(\frac{1}{9}(4 - \frac{3(n-1-i)+4)}{2^{2n-2-2i}}) + \frac{4}{3}(1 - \frac{1}{2^{2n-2i}})))$$

$$= \sum_{i=0}^{n-1}(2^{2n-2i+2}-n\cdot 2^{n-i+2}-2^{n-i+2}+i\cdot 2^{n-i+2}+2^{2n-2i+2}-2^{n-i+2}$$

$$-\frac{1}{3}\cdot 2^{2n-2i+1}+2n-2-2i+\frac{8}{3}-2^{2n-2i+1}+2)$$

$$= \sum_{i=0}^{n-1} (\frac{8}{3} \cdot 2^{2n-2i+1} - (n+2) \cdot 2^{n-i+2} + i \cdot 2^{n-i+2} + 2n-2i + \frac{8}{3})$$

$$= \frac{8}{3} \cdot 2^{2n+1} \sum_{i=0}^{n-1} \frac{1}{2^{2i}} - (n+2) \cdot 2^{n+2} \sum_{i=0}^{n-1} \frac{1}{2^i} +$$

$$+ 2^{n+2} \sum_{i=0}^{n-1} \frac{i}{2^i} + 2n^2 - 2 \sum_{i=0}^{n-1} i + \frac{8}{3}n$$

$$= \frac{8}{3} \cdot 2^{2n+1} \cdot \frac{4}{3}(1 - \frac{1}{2^{2n}}) - (n+2) \cdot 2^{n+2} \cdot 2(1 - \frac{1}{2^n})$$

$$+ 2^{n+2}(2 - \frac{n+1}{2^{n-1}}) + 2n^2 - 2 \cdot \frac{n \cdot (n-1)}{2} + \frac{8}{3}n$$

$$= \frac{1}{9} \cdot 2^{2n+6} - \frac{64}{9} - (n+2) \cdot 2^{n+3} + 8 \cdot (n+2) + 2^{n+3} - 8 \cdot (n+1) + 2n^2 - n^2 + n + \frac{8}{3} n$$

$$= \frac{1}{9} \cdot 2^{2n+6} - (n+1) \cdot 2^{n+3} + n^2 + \frac{11}{3}n + \frac{8}{9} \qquad (15)$$

The denominator of (9) can be simplified as follows:

$$\sum_{i=1}^{n} (2^i-1)^2 = \sum_{i=0}^{n} 2^{2i} - 2 \sum_{i=0}^{n} 2^i + \sum_{i=0}^{n} \cdot 1$$

$$= \sum_{i=0}^{n} 4^i - 2(2^{n+1}-1) + n+1$$

$$= \frac{4^{n+1}-1}{3} - 2^{n+2} + 2 + n + 1$$

or $\sum_{i=1}^{n} (2^i-1)^2 = \frac{1}{3}(2^{2n+2} - 3 \cdot 2^{n+2} + 3n + 8) \qquad (16)$

Substituting (15) and (16) into (9) yields

$$= \frac{\frac{1}{9} \cdot 2^{2n+6} - (n+1) \cdot 2^{n+3} + n^2 + \frac{11}{3}n + \frac{8}{9}}{\frac{1}{3}(2^{2n+2} - 3 \cdot 2^{n+2} + 3n+8)}$$

$$= \frac{\frac{16}{3} \cdot 2^{2n+2} - 6 \cdot (n+1) \cdot 2^{n+2} + 3n^2 + 11n + \frac{8}{3}}{2^{2n+2} - 3 \cdot 2^{n+2} + 3n+8}$$

$$= \frac{16}{3} - \frac{(6n-10) \cdot 2^{n+2} - 3n^2 + 5n + 40}{2^{2n+2} - 3 \cdot 2^{n+2} + 3n+8}$$

$$< \frac{16}{3}$$

Q.E.D.

It is also useful to obtain the number of nodes in the quadtree. Letting B and W correspond to the number of BLACK and WHITE, respectively, leaf nodes in the quadtree we have

Lemma 4: The number of nodes in a quadtree having B and W leaf nodes is bounded by $\frac{4}{3} \cdot (B+W)$.

Proof: See Lemma 1 in [13].

We can now prove our main result.

Theorem 3: The average execution time of the QMAT computation algorithm is of order B+W.

Proof: From Lemmas 2 and 3 we have that for each side and corner of a BLACK node, FIND_NEIGHBOR and FIND_CORNER result in an average of $4 + \frac{16}{3} = 9\frac{1}{3}$ nodes being visited. There are four sides and corners for each BLACK node. Thus these four procedures contribute $4B \cdot 9\frac{1}{3}$. From Lemma 4 we have that the number of nodes in the quadtree is bounded by $\frac{4}{3} \cdot (B+W)$. This quantity correlates with the work performed by procedure QMAT since each node in the quadtree is visited by the traversal. Summing up these values we have $4 \cdot B \cdot 9\frac{1}{3} + \frac{4}{3} \cdot (B+W) = \frac{4}{3} \cdot (29 \cdot B+W)$.

Q.E.D.

Note that our upper bound means that for small values of n the amount of work is essentially proportional to the complexity of the image--i.e., to the number of BLACK nodes.

The algorithm has an execution time complexity of the same order of magnitude as the one developed in [18] for the computation of the Chessboard distance transform--i.e., $\frac{4}{3} \cdot (43 \cdot B+W)$. Observe that an implementation of Algorithm A of Section 3 would require work proportional to B↑2 since sorting is required (a B·log B operation) as well as checking every BLACK block against the other for subsumption.

6. Concluding remarks

The concept of a skeleton and medial axis transform have been adapted to images represented by quadtrees. An algorithm for the computation of the medial axis transform for such a quadtree representation (QMAT) has been presented and shown to have an average execution time of order (B+W) where B and W correspond to the number of blocks comprising the objects and the background of the image respectively. It should be noted that the number of BLACK nodes (i.e., the image complexity) dominates the execution time of the algorithm.

The algorithm and its analysis are somewhat similar to those used in the computation of the Chessboard distance transform [18]. The difference is that for each BLACK node only its neighbors of greater or equal size needed to be examined and not their progeny as was the case in [18]. A new result of our analysis is that finding a corner neighbor is approximately 4/3 as complex as finding an adjacent neighbor. The algorithm in its present form could not be combined with the computation of the Chessboard distance transform and done in one pass (i.e., it requires a separate traversal of the tree) since computation of the QMAT relies on knowledge of the distance transform values of a node's neighbors.

The algorithm can be varied in several ways. First, in its present state, procedure QMAT overwrites the existing quadtree. It may be desirable to have an algorithm which constructs the QMAT while retaining the original quadtree. This is quite simple and can be accomplished by modifying procedure QMAT to allocate a node each time it visits one in the original quadtree. Note that our analysis assumed that all eight neighbors of a node are visited while attempting to ascertain if it is subsumed by one of its neighbors. In fact, we cease processing as soon as subsumption is found to occur. Another observation is that when overlap exists (e.g., in Figure 3b, block 10 is the NW neighbor of block 12 and is also its northern neighbor) we need not invoke FIND_NEIGHBOR or FIND_CORNER for the neighbor which overlaps the two directions. However, such a variation is of little value since the number of neighbors can be shown to range between five and eight.

The definition of the Quadtree medial axis transform in terms of the Chessboard distance transform demonstrates the appropriateness of the Chessboard distance metric for quadtrees. In particular, the analogy between squares and circles as the basis for the medial axis transform for the quadtree is noteworthy. Also, notice the similarity between the process of obtaining the QMAT and thinning [11] an image.

The advantage of the QMAT is in its compactness (e.g., recall Figures 3 and 5) and in its decreased sensitivity to shift (recall Figures 6 and 7). In the worst case, the QMAT is identical to the quadtree. The medial axis transform is often used as an alternative to a border representation because of its amenability to the determination of whether or not a given point lies within a particular region [11]. This is not a problem when the quadtree representation is used. However, in the case of the QMAT this is slightly more complex since a WHITE block does not necessarily imply that the entire space spanned by the block is WHITE--i.e., its neighbors may also have to be examined.

In Section 2 we saw that there are two ways of defining a Quadtree Skeleton with a small difference in the QMAT although the QMAT was shown to require the same number of nodes in either case. Using property (3) resulted in a simpler QMAT construction algorithm while using property (3') results in obtaining a Quadtree Skeleton of less than or equal size. Since we are primarily interested in storage compactness in the form of a tree, the difference was not important. However, we also wish to be able to reconstruct the quadtree from its QMAT. In such a case, the reconstruction process is simpler given a Quadtree Skeleton satisfying property (3') since the Quadtree Skeleton is less than or equal in size (e.g., 2 nodes vs. 3 nodes in Figures 4b and 4c respectively). In essence, a reconstruction process must add nodes corresponding to $S(t_i)$ for each $t_i \in T$, the Quadtree Skeleton. A subsequent paper will discuss the QMAT to quadtree reconstruction process in greater detail.

Frui'ful subjects for future research include the investigation of algorithms for set operations such as intersection and union as well as connectivity and perimeter using the QMAT representation. A more thorough study of the relationship between the amount of space occupied by a quadtree and its QMAT would also be welcome.

References

1. H. Blum, A transformation for extracting new descriptors of shape, in Wathen-Dunn, ed., Models for the Perception of Speech and Visual Form, M.I.T. Press, Cambridge, MA, 1967, 362-380.

2. R. O. Duda and P. E. Hart, Pattern Classification and Scene Analysis, Wiley Interscience, New York, 1973.

3. C. R. Dyer, Computing the Euler number of an image from its quadtree, Computer Science TR-769, University of Maryland, College Park, Maryland, May 1979.

4. C. R. Dyer, A. Rosenfeld, and H. Samet, Region representation: boundary codes from quadtrees, Computer Science TR-732, University of Maryland, College Park, Maryland, February 1979.

5. G. M. Hunter, Efficient computation and data structures for graphics, Ph.D. dissertation, Department of Electrical Engineering and Computer Science, Princeton University, Princeton, NJ, 1978.

6. G. M. Hunter and K. Steiglitz, Operations on images using quadtrees, IEEE Transactions on Pattern Analysis and Machine Intelligence 1, 1979, 145-153.

7. G. M. Hunter and K. Steiglitz, Linear transformation of pictures represented by quad trees, Computer Graphics and Image Processing 10, 1979, 289-296.

8. A. Klinger and C. R. Dyer, Experiments in picture representation using regular decomposition, Computer Graphics and Image Processing 5, 1976, 68-105.

9. P. Naur (Ed.), Revised report on the algorithmic language ALGOL 60, Communications of the ACM 3, 1960, 299-314.

10. J. L. Pfaltz and A. Rosenfeld, Computer representation of planar regions by their skeletons, Communications of the ACM 10, 1967, 119-122, 125.

11. A. Rosenfeld and A. C. Kak, Digital Picture Processing, Academic Press, New York, 1976.

12. H. Samet, Region representation: quadtrees from boundary codes, Computer Science TR-741, University of Maryland, College Park, Maryland, March 1979.

13. H. Samet, Computing perimeters of images represented by quadtrees, Computer Science TR-755, University of Maryland, College Park, Maryland, April 1979.

14. H. Samet, Connected component labeling using quadtrees, Computer Science TR-756, University of Maryland, College Park, Maryland, April 1979.

15. H. Samet, Region representation: raster-to-quadtree conversion, Computer Science TR-766, University of Maryland, College Park, Maryland, May 1979.

16. H. Samet, Region representation: quadtrees from binary arrays, Computer Science TR-767, University of Maryland, College Park, Maryland, May 1979.

17. H. Samet, Region representation: quadtree-to-raster conversion, Computer Science TR-768, University of Maryland, College Park, Maryland, June 1979.

18. H. Samet, A distance transform for images represented by quadtrees, Computer Science TR-780, University of Maryland, College Park, Maryland, July 1979.

19. M. Shneier, A path-length distance transform for quadtrees, Computer Science TR-794, University of Maryland, College Park, Maryland, July 1979.

20. M. Shneier, Linear-time calculations of geometric properties using quadtrees, Computer Science TR-770, University of Maryland, College Park, Maryland, May 1979.

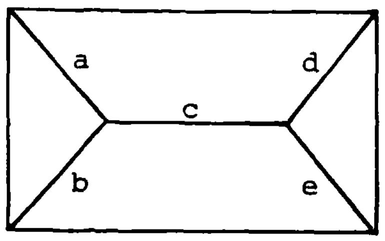


Figure 1. A rectangle and its skeleton using d_E.

```
1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1
```

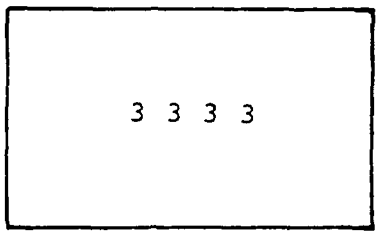
a. Image

```
0                 0
  1             1
    2         2
      3 3 3 3
    2         2
  1             1
0                 0
```

c. MAT of the image in (a) using d_A.

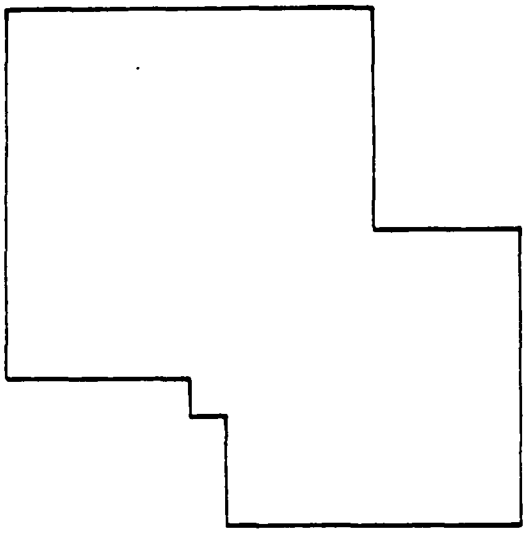
```
1 1 1 1 1 1 1 1 1 1
1 2 2 2 2 2 2 2 2 1
1 2 3 3 3 3 3 3 2 1
1 2 3 4 4 4 4 3 2 1
1 2 3 3 3 3 3 3 2 1
1 2 2 2 2 2 2 2 2 1
1 1 1 1 1 1 1 1 1 1
```

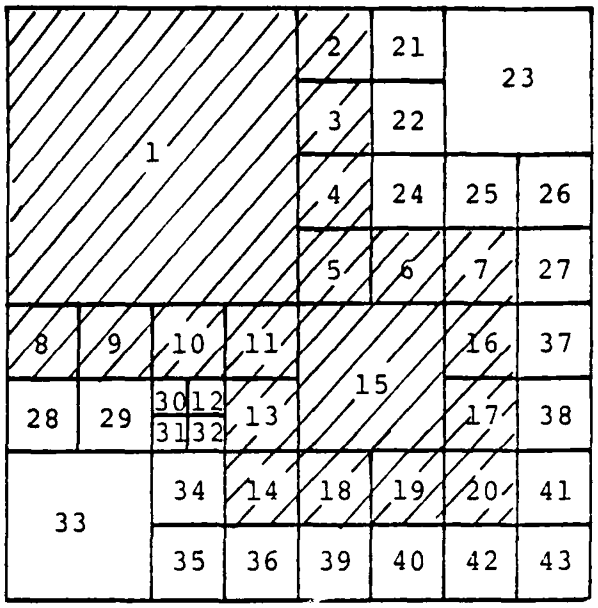
b. d_A and d_M for the image in (a).



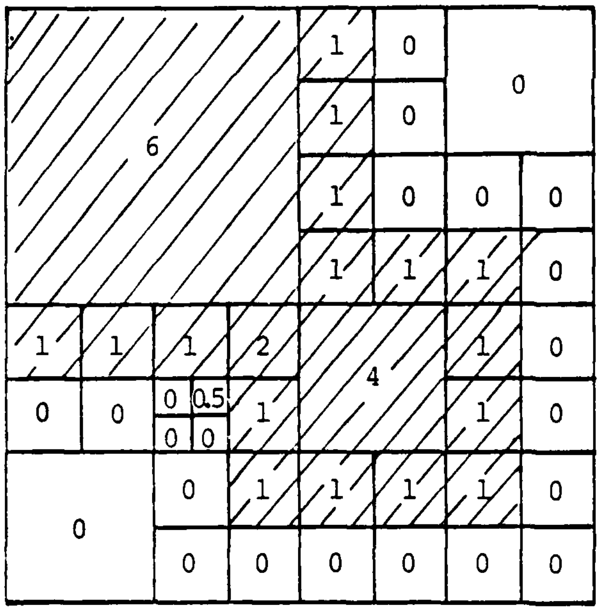
d. MAT of the image in (a) using d_M.

Figure 2. A binary array representation of an image, its distance transform, and its MAT.

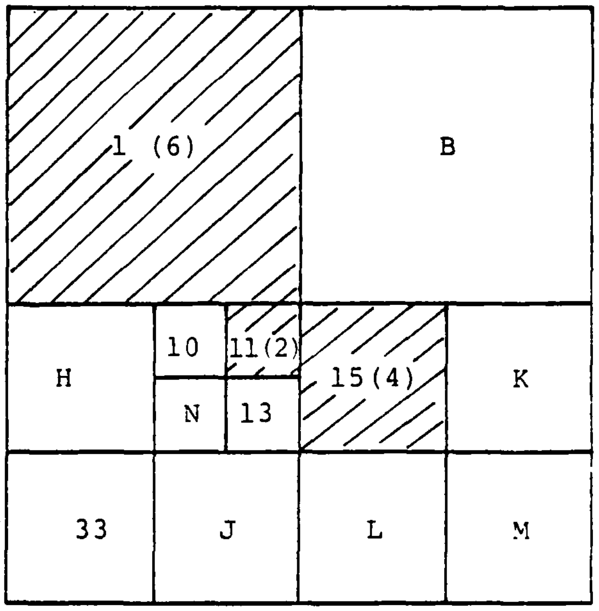

a. Sample image.



b. Block decomposition of the image in (a).

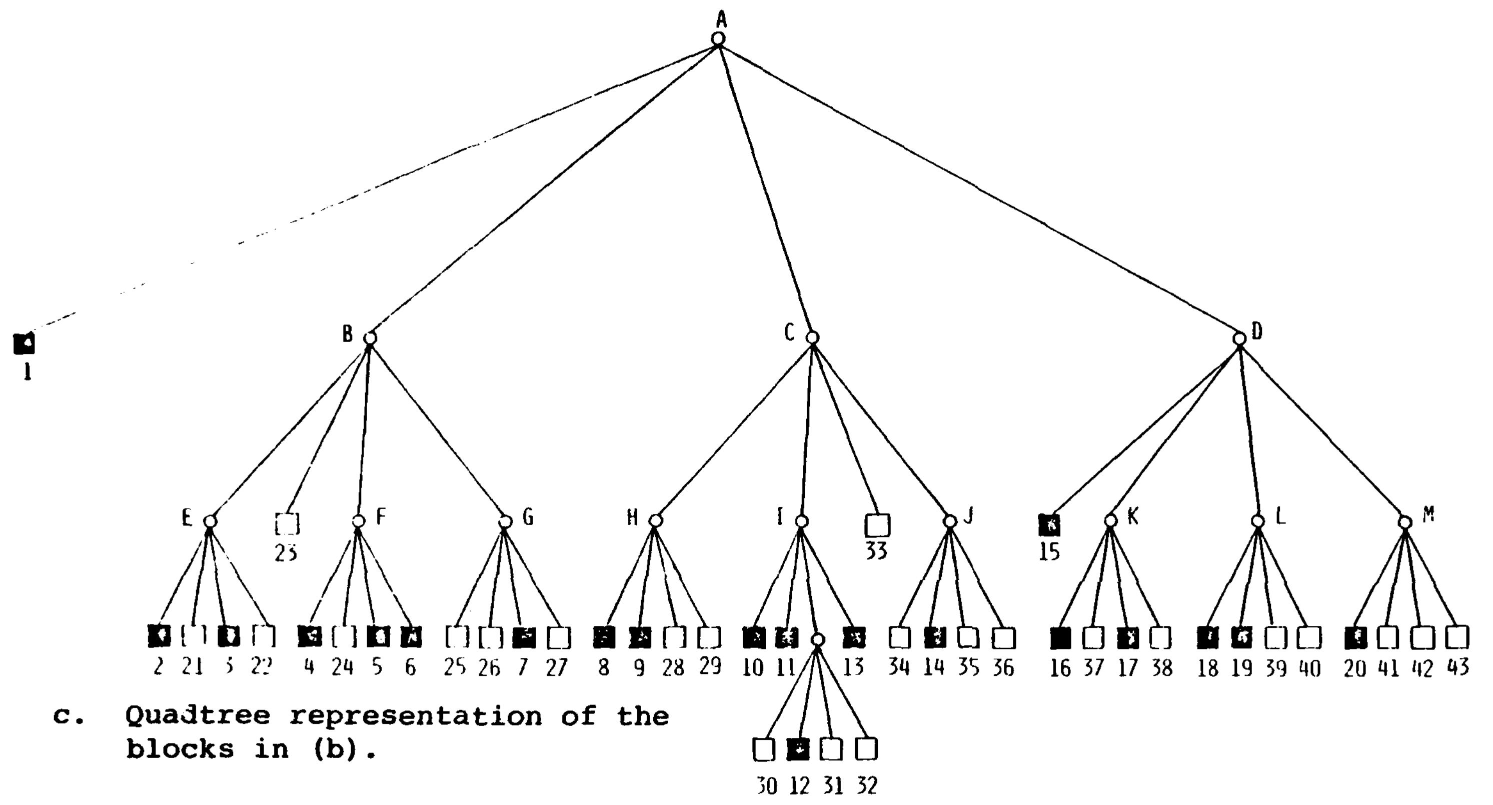


d. Chessboard distance transform of (b).



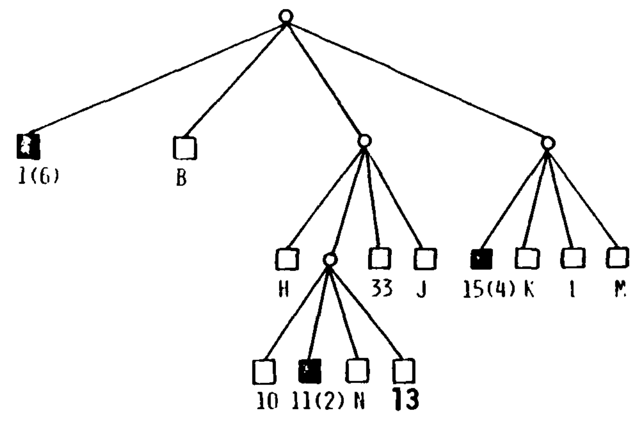
e. Block decomposition of the QMAT of (b). Radius values are within parentheses.

Figure 3. An image, its maximal blocks, the corresponding quadtree, the chessboard distance transform, the block decomposition of the QMAT, and the QMAT. Blocks in the image and in the QMAT are shaded.

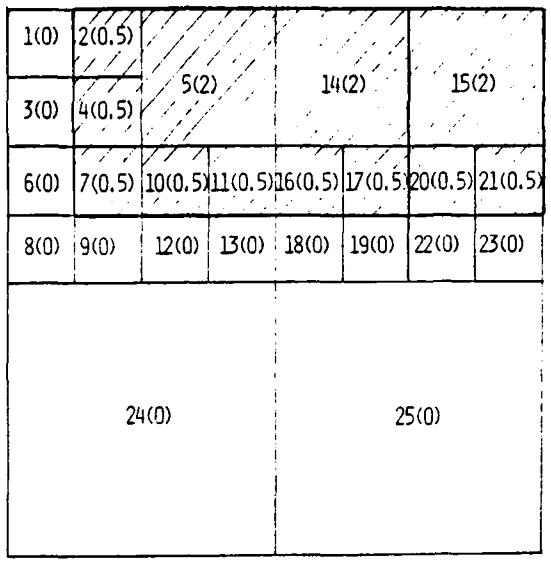


c. Quadtree representation of the blocks in (b).

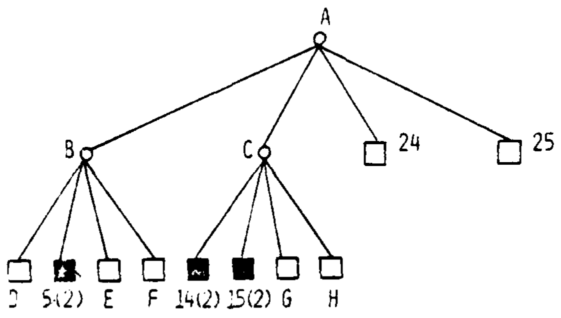
Figure 3 (continued)



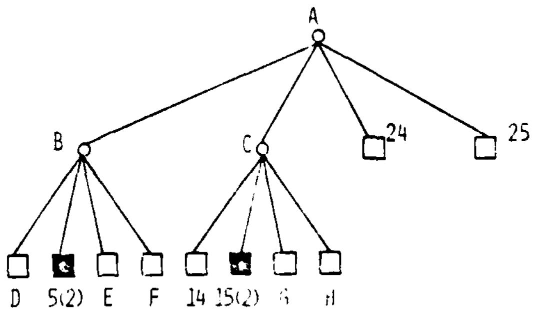
d. QMAT representation of the blocks in (b). Radius values are within parentheses.



a. Image. The value of the Chessboard distance transform is within parentheses.

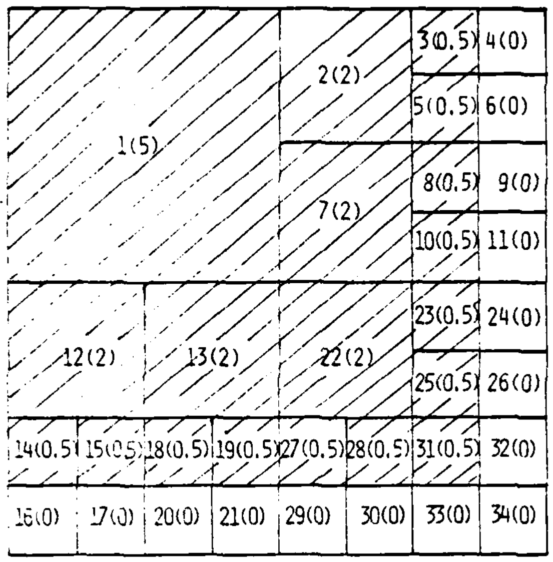


b. QMAT of the image in (a) using property (3). Radius values are within parentheses.

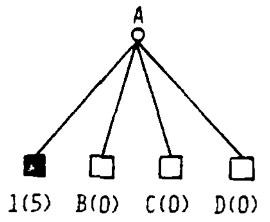


c. QMAT of the image in (a) using property (3'). Radius values are within parentheses.

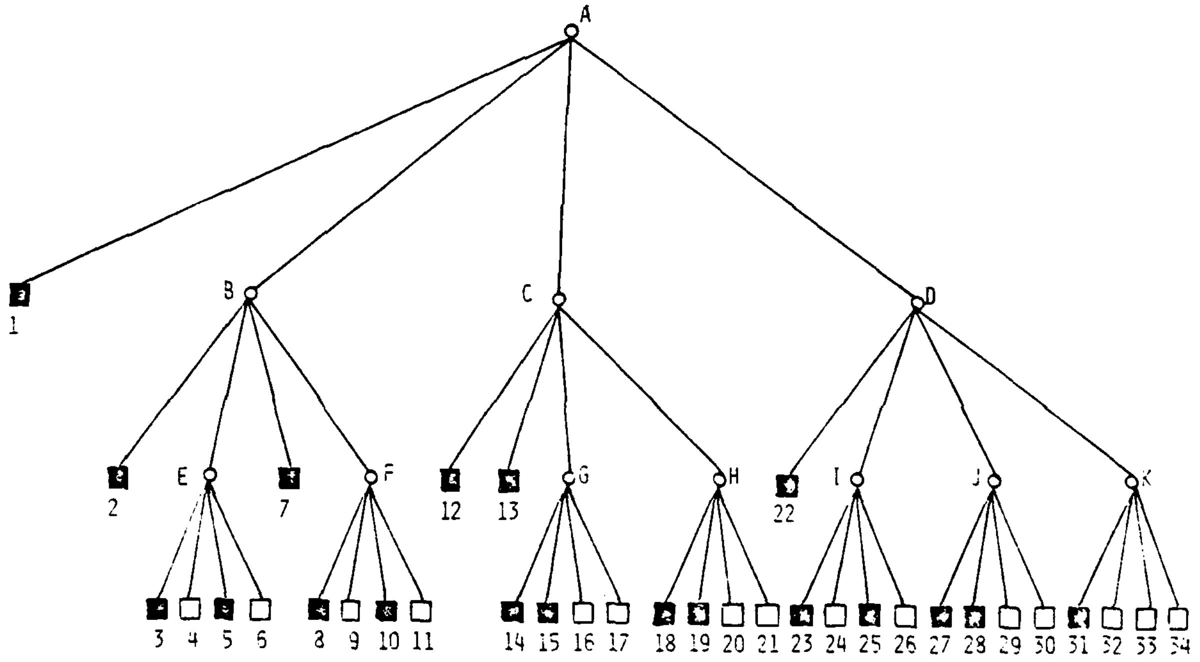
Figure 4. An image and its corresponding QMATs using properties (3) and (3') for the Quadtree Skeleton definitions. Blocks in the QMAT are shaded.



a. Image. The value of the Chessboard distance transform is within parentheses.

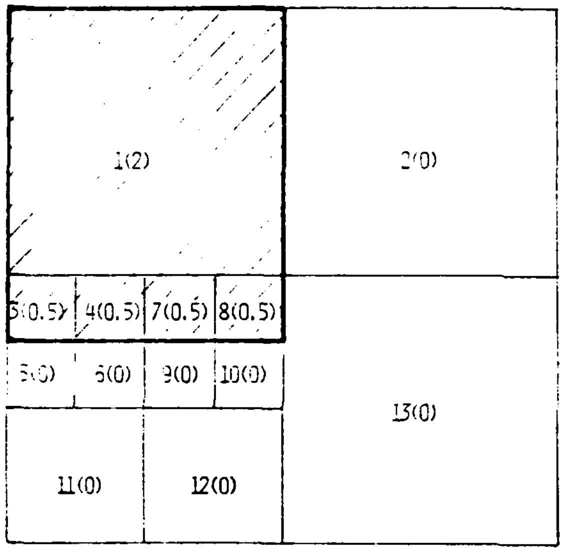


c. QMAT representation of the image in (a). Radius values are written within parentheses.

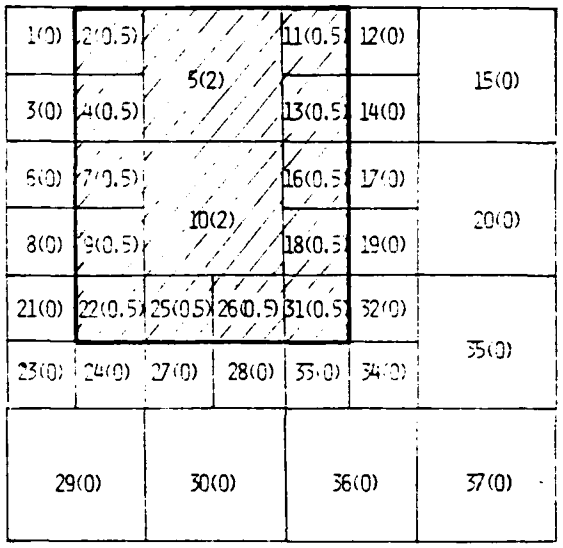


b. Quadtree representation of the image in (a).

Figure 5. An image and its corresponding quadtree and QMAT illustrating the maximum compactness that can be achieved as a result of using the QMAT as a data structure. Blocks in the image are shaded.

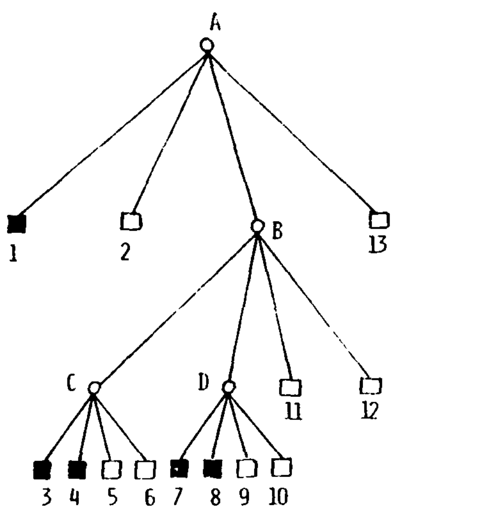


a. Image. The value of the Chessboard distance transform is within parentheses.

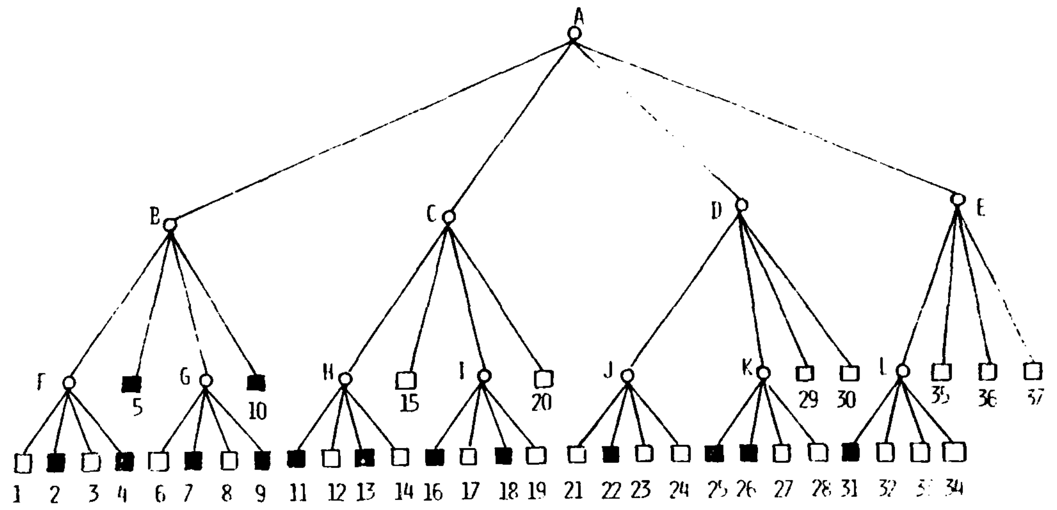


d. The image in (a) shifted by one unit to the right. The values of the Chessboard distance transform is within parentheses.

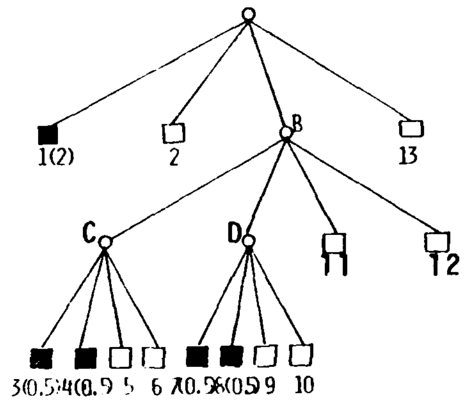
Figure 6. An image and its corresponding quadtree and QMAT, and the result of shifting it by one unit to the "right". Blocks in the image are shaded.



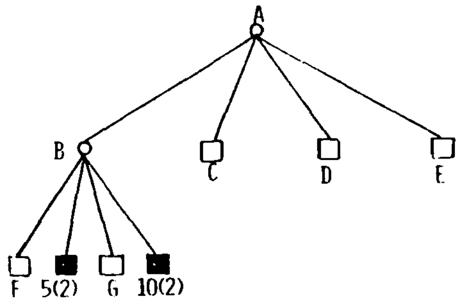
b. Quadtree representation of the image in (a).



e. Quadtree representation of the image in (d).

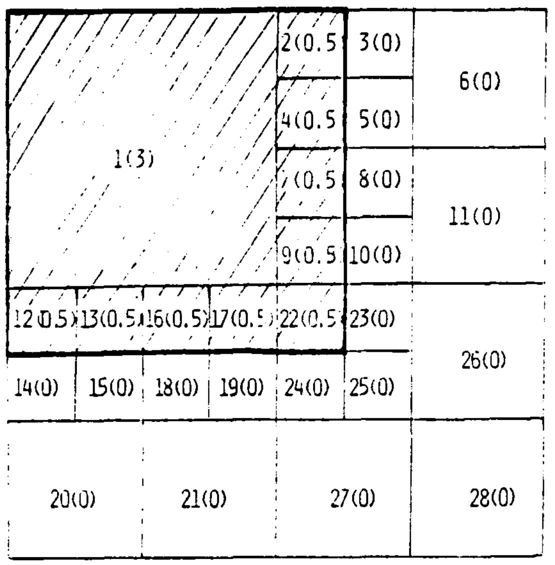


c. QMAT representation of the image in (a). Radius values are within parentheses.

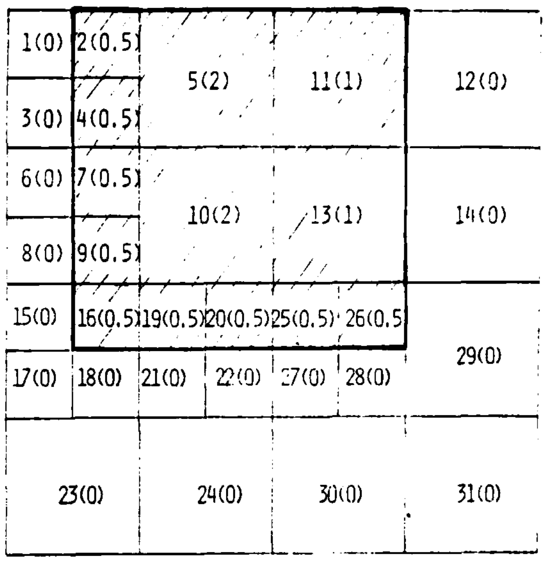


f. QMAT representation of the image in (d). Radius values are within parentheses.

Figure 6 (continued)

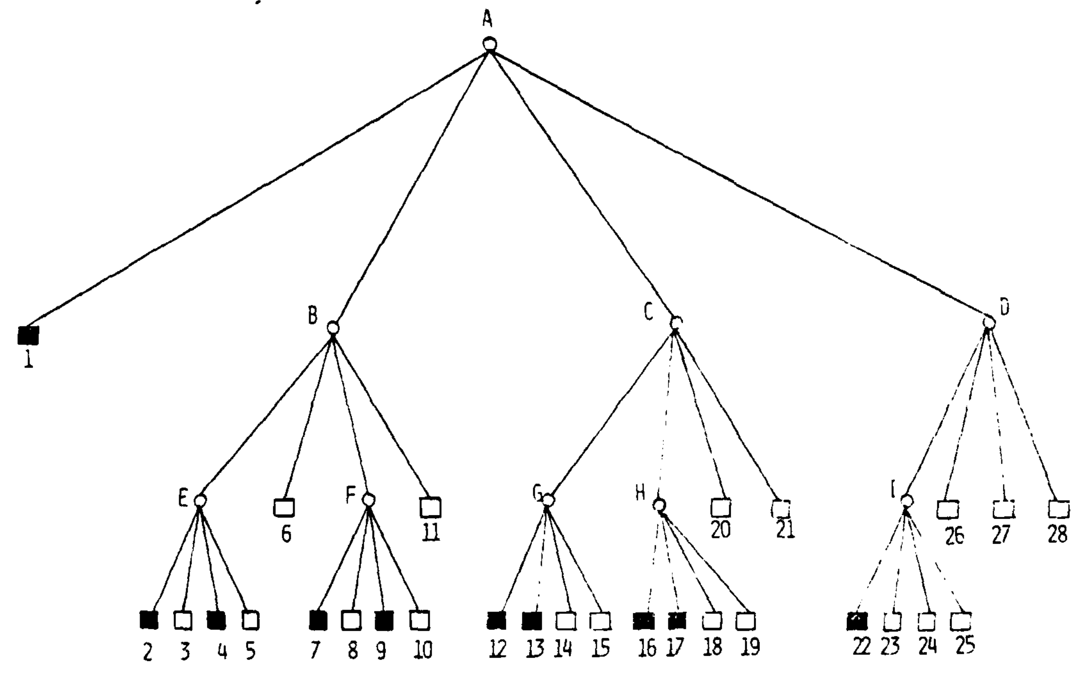


a. Image. The value of the Chessboard distance transform is within parentheses.

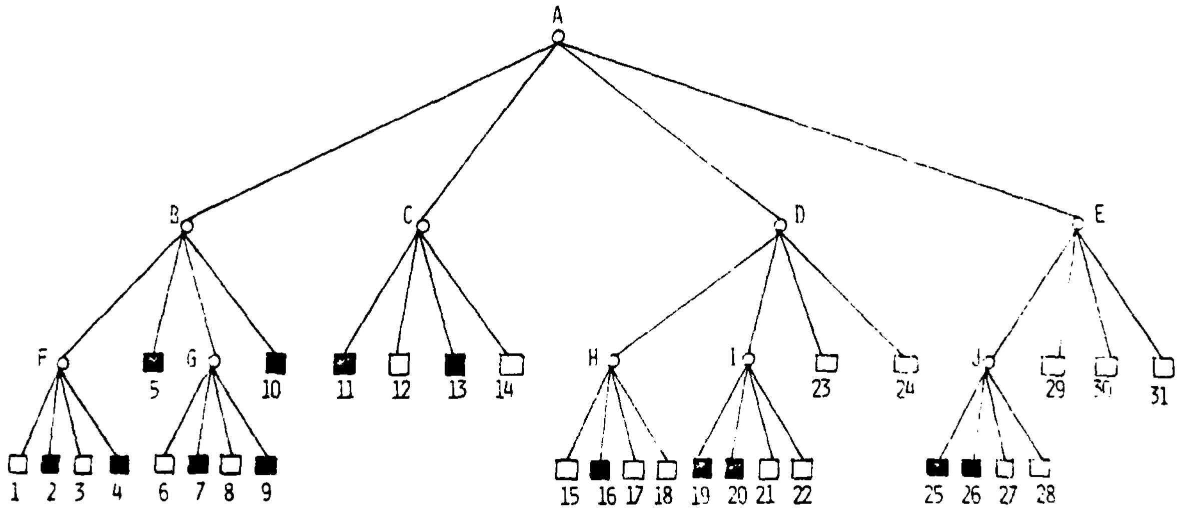


d. The image in (a) shifted by one unit to the right. The value of the Chessboard distance transform is within parentheses.

Figure 7. An image having a minimal QMAT and its corresponding quadtree and QMAT, and the result of shifting it by one unit to the right. Blocks in the image are shaded.

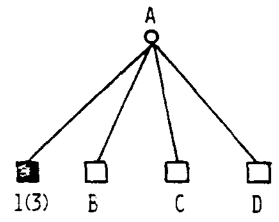


b. Quadtree representation of the image in (a).

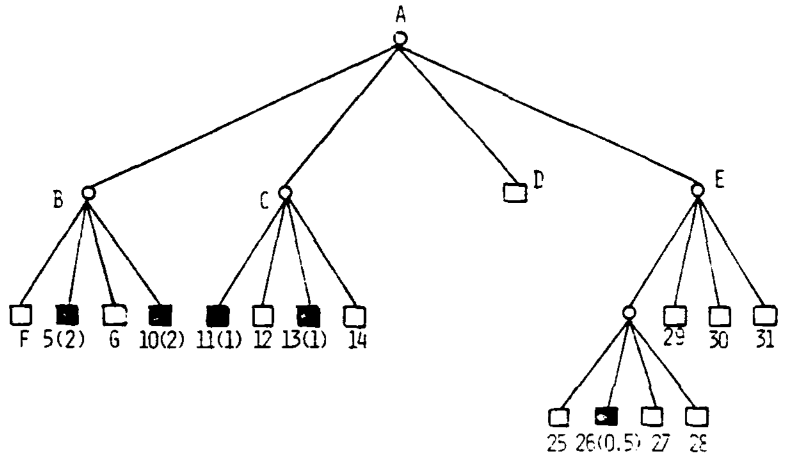


e. Quadtree Representation of the image in (d).

Figure 7 (continued)



c. QMAT representation of the image in (a). Radius values are within parentheses.



f. QMAT representation of the image in (d). Radius values are within parentheses.

Figure 7 (continued)

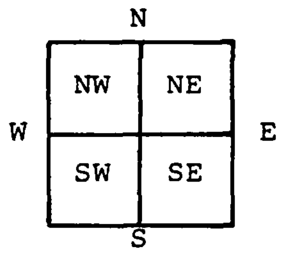


Figure 8. Relationship between a block's four quadrants and its boundaries.

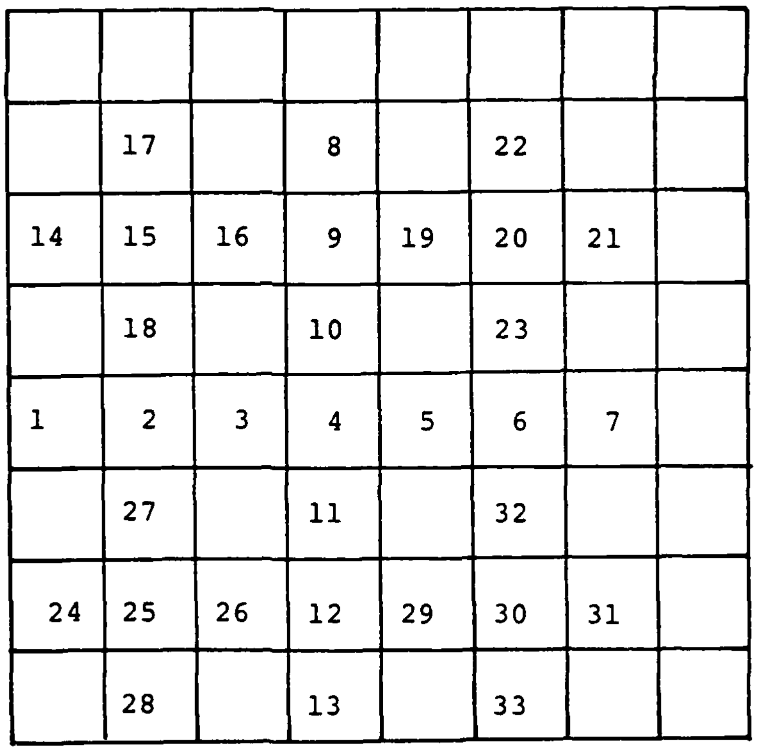


Figure 9. Sample grid illustrating blocks whose nodes are at level Ø and whose nearest common ancestor is at level ≥2 when attempting to locate a NE neighbor.

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A086 097	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A QUADTREE MEDIAL AXIS TRANSFORM		5. TYPE OF REPORT & PERIOD COVERED Technical
		6. PERFORMING ORG. REPORT NUMBER TR-803
7. AUTHOR(s) Hanan Samet		8. CONTRACT OR GRANT NUMBER(s) DAAG-53-76C-0138
9. PERFORMING ORGANIZATION NAME AND ADDRESS Computer Science Department University of Maryland College Park, MD 20742		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS U.S. Army Night Vision Lab. Fort Belvoir, VA 22060		12. REPORT DATE August 1979
		13. NUMBER OF PAGES 51
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Image processing	Quadtrees
Pattern recognition	Medial axis transforms
Region representation	Skeletons

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The skeleton and medial axis transform concepts used in traditional image processing representations are adapted to the quadtree representation. A new data structure termed the Quadtree Medial Axis Transform (QMAT) is defined. An algorithm is presented for the computation of the QMAT of a given quadtree by only examining each BLACK node's adjacent and abutting neighbors. Analysis of the algorithm reveals an average execution time proportional to the

number of leaves in the quadtree. Some of the interesting properties of the QMAT vis a vis the quadtree are its compactness and a decreased shift sensitivity.